Digital Systems (Sheet #5)

Marius Gavrilescu

15.1 Without buffering, the time taken for n outputs is $np + nw + nc$. With buffering, the time taken is $np + w + nc$.

With double buffering, a buffer is filled in np and emptied in $w + nc$. We can fill the $(i + 1)th$ buffer at the same time we're emptying the ith buffer, so for $m + 1$ buffersof data the time taken is $np + w + nc + m * max(np, w + nc)$.

With triple buffering we won't be able to increase the throughput. One reason to use triple buffering is to ensure both the consumer and the producer can get full or respectively empty buffers with little blocking. This is useful for time-sensitive applications such as computer graphics.

15.2 Producing the output can be significantly slower than printing it. By spooling we ensure the printer does not have to wait for more data to be produced, which simplifies its design.

15.3 Simple example: requests are made for tracks at distances 20, 10, and 5 from the head. Serving them in the order in which they are made takes 20 + 10 + 5 = 35 units of time, while serving them in order of shortest seek first takes only 20 units of time.

We could have a request for a far away track and then many requests for close tracks. As long as requests for tracks close to the head keep coming in, the far away request will never be served.

15.4 It could either wait for the sector to come under the head or continue serving requests in the direction of movement and serve this request on the way back. Waiting for the sector to come could create the same problem that "shortest seek first" has, since requests could keep coming for sectors on this cylinder and the head will never move to another cylinder.

The order is 5, 3, 1 (moving down), then 7, 9, 11 (moving up).

15.5

16.1 Processing the data that was just read could take some time. By keeping some space between consecutively numbered sectors we have some processing time between every two consecutive sectors thus allowing us to read many consecutive sectors and processing them on-the-fly (without needing a buffer). A disadvantage is that if the processing time is short (e.g. we have a buffer) it will take longer to read consecutive sectors.

Since platters are spinning (almost) constantly, during the time the head is moving from cylinder n to cylinder (n+1) the head will be on a different sector. If we have a sequential request spanning multiple consecutive cylinders we might want to skew the first sector of every cylinder such that when we move the head from the last sector of cylinder n to cylinder (n+1), it will point to the first sector of cylinder (n+1).

17.3
```
-- Copyright (C) 2016 Gabriel-Robert Inelus
sema nfree = n;
sema mutex = 1;
sema mutex2 = 1;
bool free[n]; -- initially all true

proc allocate(var i)
  wait(nfree)
  wait(mutex)
  for(int j = 0; j < n; ++j)
    if(free[j] == true){
       i = j;
       free[j] = false;
       break;
    }
  signal(mutex);

proc release(i)
  wait(mutex2)
  free[i] = true;
  signal(nfree)
  signal(mutex2)
```